



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/764,526	01/17/2001	Christopher R. Sheedy	20206-39 (P00-3337)	6417

7590 05/19/2004

HEWLETT-PACKARD COMPANY
Bill Streeter
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400

EXAMINER

RUTTEN, JAMES D

ART UNIT	PAPER NUMBER
----------	--------------

2122

DATE MAILED: 05/19/2004

9

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

09/764,526

Applicant(s)

SHEEDY, CHRISTOPHER R.

Examiner

J. Derek Rutten

Art Unit

2122

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 24 February 2004.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-17 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-17 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
 2. ☐ Certified copies of the priority documents have been received in Application No. _____.
 3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
- * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. Applicants' amendment dated February 24, 2004, responding to the November 26, 2003 Office Action provided in the rejection of claims 1-13, wherein claims 1-3, 7, and 9-13 have been amended. Claims 14-17 have been added. Claims 1-17 remains pending in the application and have been fully considered by the examiner.

Applicant's arguments with respect to the rejection of the claims have been considered but are moot in view of the new grounds of rejection.

2. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than **SIX MONTHS** from the date of this final action.

3. Claims 1, 2, and 7 are finally rejected as being unpatentable over Boehm, Evans and Tanaka. Claim 3 is finally rejected as being unpatentable over the combination of Boehm, Evans, and Tanaka in view of Shute. Claims 4-6 are finally rejected as being unpatentable over

Art Unit: 2122

the combination of Boehm, Evans, Tanaka, and Shute in view of Schach. Claim 8 is finally rejected as being unpatentable over the combination of Boehm, Evans, and Tanaka in view of Leblang. Claims 9 and 10 are finally rejected as being unpatentable over the combination of Boehm, Evans, and Tanaka in view of Hunt. Claims 11-13 are finally rejected as being unpatentable over the combination of Boehm, Evans, Tanaka, and Hunt in view of Schach. Claims 14-17 are finally rejected as being unpatentable over Evans in view of Tanaka.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

5. Claims 14-17 are rejected under 35 U.S.C. 101 because the disclosed invention is inoperative and therefore lacks utility. Claim 14 claims "A method comprising: creating source code...compiling the source code...building an executable program..." While these process steps could be useful if they are claimed as a computer implemented method, they are currently claimed as abstract steps that do not provide utility since there is no functional relationship in the absence of implementation. It is suggested that claim 14 recites "A computer implemented method comprising..."

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

Art Unit: 2122

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

7. Claims 1, 2, and 7 are rejected under 35 U.S.C. 103(a) as being unpatentable over prior art of record U.S. Patent 6,457,170 to Boehm et al. (hereinafter referred to as "Boehm") in view of prior art of record U.S. Patent 5,805,899 to Evans et al. (hereinafter referred to as "Evans") further in view of U.S. Patent 6,665,735 to Tanaka et al. (hereinafter referred to as "Tanaka").

As per claim 1, Boehm discloses:

A computerized method of saving version and product information of a library in an executable program, comprising: creating a version source file, the version source file containing version information and product information of the library; [Boehm's invention has taught us a method and apparatus for building a software system, in SUMMARY, line 43-46, "the invention lists each source file, by name and version number, (the name can be the product information) and each explicit object file (by name) that comprise the software system to be built and form that build list..." Line 50-52, "Source file links associate source file names and versions listed on the build list (here the build list is same as the version source file stated in claim 1 first item) to a copy of the corresponding source file store in the network cache memory".]

compiling the version source file to create a version object file; rebuilding the library to include the version object file; [In Boehm's column 13, line 1-3, "the source file handler 207 generates the potentially usable object filename 'Adder.o' from the source file 'Adder.src,'" this shows the version source file has been compiled into the version object file as recited in second

Art Unit: 2122

item of claim 1. It also mentioned, "rebuild object files that may be changed from the prior builds" (see Boehm's Abstract). - In order to make a new executable, the library would have to be rebuilt with the newly compiled object code.] *and*

building the executable program such that the version information and product information of the library is combined into the executable program. [In Boehm's column 8 line 35-40, "After the object file handler processes the explicit object files listed on the build list 100 and the potentially usable object files generated by the source file handler, the program can be built using make or another software-building program (shown as step 300 in Fig. 2)." The make tool compiles given source files to object files and link them all together to form an executable program therefore all the version and product information (file name, date and time of the build, build name...) is combined into the executable program.]

Boehm does not expressly disclose: *a version function whose name comprises at least one of version information and product information.*

However, in an analogous environment, Evans teaches the feature of a method and apparatus for providing a mapfile specifying a function name associated with a version of the software program. Evans' invention taught us that the global symbols and version names associated with each version are defined in a mapfile (Evans column 2, line 26-27). Here the function symbol is the global symbol, which is a symbol of a function that is known globally in the program.

Also in an analogous environment, Tanaka teaches naming a function solely for the purpose of conveying information from a library [column 6 lines 64-66: "Because the dummy

Art Unit: 2122

function func2 is needed only to describe the symbol name “func2” in the LIB file 31, it can be empty.”].

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the name reference technique of Tanaka with Evans' version names in Boehm's software development environment. One of ordinary skill would have been motivated to link a library containing a function that describes another library to permit the explicit specification of the version of an object to which the software program should be linked.

As per claim 2, the above rejection of claim 1 is incorporated. Further, Boehm discloses:

wherein creating the version of the source file further comprises:

constructing a version string, the version string containing version and product identification information pertaining to the library [in Boehm's SUMMARY, lines 43-46, the invention lists each source file, “by name and version number”. Here, the name can include product identification information. For example, the file name Boehm used in rejection of claim 1 (2), *adder.src* can be modified to *adder 17 T 4 2003 v1.ada*, which shows the function of the file (it's an adder), the date (1 111412003), the version (1), and the file is a source file in 'Ada' programming language. The version string can include all the product identification information that the user needs.];

combining function symbols with the version string to form the version function

[Boehm teaches all aspects of the applicant's claims except it does not specifically mention to associate function symbols with version name. However, Evans teaches the

feature of a method and apparatus for providing a mapfile specifying a function name associated with a version of the software program. Evans' invention taught us that the global symbols and version names associated with each version are defined in a mapfile (Evans column 2, line 26-27). Here the function symbol is the global symbol, which is a symbol of a function that is known globally in the program. Therefore, it would have been obvious to a person of the ordinary skill in the art at the time of the invention to modify Evans' system with the source file versioning feature ("mapfile") for the same reason it is taught by Boehm, to provide a build list that associates the version information with the source file and based on the version information to produce versioned object and, further to build versioned executable program.];

creating a version source file whose name includes a keyword and the name of the at least one library [See the rejection of claim 1; the build list name can include a keyword and the name of the library.]; *and*

storing the version function in the version source file [Boehm's mapfile can include the version function name. It has to be stored in the version source file in order to be built in the executable program.].

As per claim 7, in Boehm's invention, column 1, lines 25-29, "... efficient software design, by providing a **library or storehouse** of previously developed software items and modules that can be re-used, modified, expanded, or reconfigured as required to support development of upgrades, new applications or new systems." Boehm discloses

Art Unit: 2122

a library or a storehouse, which is used to reconfigure libraries as required, here the reconfiguration can be **removing** or **remaking** a library.

8. Claim 3 is rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Boehm, Evans, and Tanaka as applied to claim 1 above, further in view of prior art of record “Separate Compilation and the UNIX make Program” by Shute (hereinafter referred to as “Shute”).

As per claims 3, Boehm discloses:

constructing a version string, the version string containing version and product identification information pertaining to the library (See above rejections of claims 1 and 2);

combining a build identifier and function symbols with the version string to form a name of the version function [See rejection of claim 2. Boehm has mentioned using existing build tool, for example, the *make* tool (rejection of claim 1). Boehm has taught us to use the make tool, but didn't mention the detail of how to specify source file name, object file name and assign a build name. The make command is described in detail in “Separate Compilation and the UNIX make Program” (by Gary Shute, 09/23/99). Page 4 of this reference shows the way of specifying these names. The example was given on page 4: “*table.o: table.c table.h*” where to enter source file names (table.c and table.h) and to produce an object file (table.o), and to create a build by “*calculator: calculator.o table.o*” (using object files calculator.o and table.o to create a build named *calculator*).

Note that any function symbols or version strings can also be used here; the user can combine a build identifier and function symbols with the version string to form a build-identified version function that the user wants to be included in the new build.

It would have been obvious to a person of the ordinary skill in the art at the time of the invention to utilize the makefile concept for the same reason it is taught by Boehm, to combine a build identifier and function symbols with the version string to create a version source file.];

creating a version source file whose name includes a keyword and the name of the library; and storing the version function in the version source file [See rejection of claim 2].

9. Claims 4-6 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Boehm, Evans, Tanaka, and Shute as applied to claim 3 above, and further in view of prior art of record "Software Engineering" by Schach (hereinafter referred to as "Schach").

As per claim 4, Boehm teaches all aspects of the applicant's claims except he does not specify how to name the build identifier.

However, Schach teaches the feature of what to be included in a build name so it can uniquely identify builds In Schach's "Software Engineering", section 11.5 1, page 353, 3rd paragraph, "... a detailed record (or derivation) of every version of the product must be kept. This comprises the name of each source code element, including the

Art Unit: 2122

variation and revision (as recited in claim 5, build identifier), the versions of the various compilers and linker used, the name of the person (as recited in claim 6, a user that performs the build) who constructed the product, and, of course, the date and the time (as recited in claim 4, date) at which it was constructed." This paragraph basically covers claims 4, 5, and 6.

It would have been obvious to a person of the ordinary skill in the art at the time of the invention to include Schach's idea about uniquely identify builds for the same reason it is taught by Boehm, to provide a build list that associates the version information with the source file and based on the version information to produce versioned object and further to build versioned executable program.

As per claims 5 and 6, all limitations have been addressed in the above rejection of claim 4.

10. Claims 9 and 10 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Boehm, Evans, and Tanaka as applied to claim 1 above, and further in view of prior art of record U.S. Patent 6,499,137 to Hunt (hereinafter referred to as "Hunt").

As per claim 9, list, Boehm teaches all aspects of the applicant's claims about creating build compiling, and rebuilding (see rejection of claim 1) executable program, but Boehm does not specifically mentioned his invention is for 'a plurality of libraries'.

However, both Evans and Hunt teach the idea of including multiple libraries (object files) in a new build.

In Evans' invention, Column 1, 2nd paragraph, "The software development process is especially problematic when a software application binds to **shared objects (also called 'libraries')**." Evans described the 'plurality of object files' as 'plurality of libraries'. In Hunt's invention. Abstract 3rd line, "... a user selects a library for linking to the compiled application. An association is made between the **selected library and any external libraries** referenced within the compiled application."

It would have been obvious to a person of the ordinary skill in the art at the time of the invention to modify Evans' and Hunt's including a plurality of libraries for the same reason it is taught by Boehm, to provide a build list that associates the version information with the source file (from the plurality of libraries) and based on the version information to produce versioned object files, and further to build a versioned executable program.

As per claim 10, the above rejection of claim 9 is incorporated.

selecting from the plurality of libraries a group of libraries needed for the building of the executable, each library in the group having a version object file [See the rejection of claims 1 and 9]; and wherein building the executable includes:

*creating a temporary storage area [In Boehm's invention, column 1, lines 25-29, "... efficient software design, by providing a **library or storehouse** of previously*

Art Unit: 2122

developed software items and modules that can be re-used, modified, expanded, or reconfigured as required to support development of upgrades, new applications or new systems.” Here a **library or a storehouse** is used as a **temporary storage area** for manipulating new builds.];

obtaining the version object file from each of the selected libraries, each version object file having a name that includes a keyword and the name of the library in which the version object file resides [See rejections of claims 1, 2, and 9];

storing each of the version object files in the temporary storage area [See rejection of claim 2, and Boehm column 1 lines 25-29 as cited above];

creating a list of the names of the stored version object files [See rejection of claim 1]; *and*

compiling into the executable each of the stored version object files in the list so that the executable contains any functions needed by the executable from each library in the group and the version and product information of each library of the group [See rejection of claims 1 and 9].

11. Claim 8 is rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Boehm, Evans, and Tanaka as applied to claim 1 above, and further in view of prior art of record U.S. Patent 5,649,200 to Leblang.

As per claim 8,

wherein building the executable includes:

creating a temporary storage area [In the rejection of claim 10, Boehm teaches the idea of keeping a 'storehouse' as a working space, where old space may be removed, and new space can be allocated (reconfigured); even Boehm's library or storehouse has the same function as a temporary storage area.

Leblang also shows similar ideas, in column 12, lines 18-22. "View-private storage ensures that developers have enough 'elbow room' to work effectively without getting in each others way." Here the **elbow room** is used as a **temporary storage space**.

It would have been obvious to a person of the ordinary skill in the art at the time of the invention to include Leblang's idea about keeping 'elbow room' to make work effectively for the same reason it is taught by Boehm to provide enough space, to reconfigure the space as required to support development of upgrades, new application, or new systems.];

All further limitations have been addressed in the above rejections of claims 1 and 2 as cited in the previous Office action.

12. Claims 11-13 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Boehm, Evans, Tanaka, and Hunt as applied to claim 9 above, and further in view of Schach.

As per claim 11, Boehm teaches the entire flow of selecting group libraries, building the object files, which includes version source files, and build executable with

Art Unit: 2122

the entire versioned object files. But Boehm does not specifically mention building a compound library.

However, in Schach's "Software Engineering", section 11.6.5, page 359, 3rd paragraph, "For each executable load image, the programmer sets up a *Makefile* specifying the hierarchy of source and object files that go into that particular configuration; such a hierarchy is shown in Figure 11.2." Figure 11.2 on page 352 is very important, it shows the way selecting different groups of libraries (object files), binding object files into a compound library (executable load image), and the new executable program is stored within the compound library. Schach teaches the feature of combining various libraries into a compound library. The UNIX *make* program is an existing build tool which builds new executable program from given versions of source file/object files within different libraries.

It would have been obvious to a person of the ordinary skill in the art at the time of the invention to include Schach's idea about making an executable from the compound library and using the make command, for the same reason it is taught by Boehm, to provide a build list that associates the version information with the source file, and based on the version information to produce versioned object files, and further to build versioned executable program.

As per claim 12, the above rejection of claim 11 is incorporated. All further limitations have been addressed in the above rejections of claims 1, 2, 9, and 10 as cited in the previous Office action.

As per claim 13, all further limitations have been addressed in the above rejections of claims 1, 2, 9, and 11 as cited in the previous Office action.

13. Claims 14-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Evans in view of Tanaka.

As per claim 14, Evans discloses:

A method (column 14 line 53 – column 15 line 41) comprising:

creating source code file containing version information of a library (column 2 lines 25-28: “In the described embodiment of the invention, the global symbols and version names associated with each version are defined in a ‘mapfile’ generated by a human being.”);

*compiling the source code file to create an object file placed within the library (column 2 lines 8-11: “At **build time**, a link-editor adds data to a versioned object defining all available versions of the object (a version definition section and a version symbol section).”); and*

building an executable program using at least a function from the library such that the version information is contained in the executable program (column 2 lines 30-34: “By default, at build time, when an application is link-edited with a versioned object that contains versioning information, a dependency is established in the application to those versions that include the global symbols referenced by the application.”).

Art Unit: 2122

Evans does not expressly disclose a function whose name comprises version information, the version information thus contained in the executable program through the presence of the function.

However, in an analogous environment, Tanaka teaches the technique of including a function in a library as a way to describe information relating to another library (column 6 lines 63-66: "The dummy library 34 (DLL2) includes a dummy function func2. Because the dummy function func2 is needed only to describe the symbol name "func2" in the LIB file 31, it can be empty.").

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Tanaka's dummy function library in Evan's versioning system. One of ordinary skill would have been motivated to avoid renaming an object, or library, each time a new version of the object is created.

As per claim 15, the above rejection of claim 14 is incorporated. Evans does not expressly disclose a function with an empty body.

However, Tanaka teaches the use of a function with an empty body (column 6 lines 63-66 as cited above in the rejection of claim 14). A function with an empty body is inherently a void function since there would be no return type associated with that function.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Tanaka's empty function in Evan's versioning system. One of

Art Unit: 2122

ordinary skill would have been motivated to describe an associated symbol name in a library file so that function pre/post processing can occur.

As per claim 16, the above rejection of claim 14 is incorporated. Further, Evans discloses a non-void function with text in the body (column 6 lines 58-60).

As per claim 17, the above rejection of claim 14 is incorporated. Further, Evans discloses a function name with both product and version information (column 7 line 6).

Art Unit: 2122

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to J. Derek Rutten whose telephone number is (703) 605-5233. The examiner can normally be reached on M-F 6:30-3:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

jdr



TUAN DAM
SUPERVISORY PATENT EXAMINER